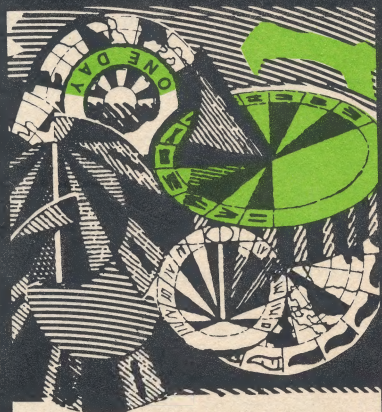


SoftSideTM Selections



it's about time

A product of **SoftSide** Publications, Inc.

6 South Street, Milford, NH 03055.

Index



FRONT RUNNER: Flip-It II by Michael Prescott.

TRS-80 version by Alan J. Zett.

In this board game, you find your computer a formidable opponent as you match wits trying to outflank and capture one another's pieces. 1



DV Bonus Program: Force of Arms by Stephen J. Choi

Can you conquer all of the Earth's territories, and dominate the globe? Only the craftiest general can overcome his adversaries in this war of cunning, strategy, chance, and **Force of Arms**. 9



Savo Island by Victor A. Vernon, Jr.

The WW II Battle of Savo Island was a humiliating Allied defeat. In this computer simulation, however, you have a chance to change the history books. 12



Adventure Disk and Cassette Bonus

SoftSide Adventure Series: It's About Time

by Peter Kirsch.

Strolling in your neighborhood one day, you stumble upon a Time Machine. It propels you into the far future, where Henry Bowman's B Bomb has annihilated all you knew. Is Earth doomed to hellfire? 27

General Information 28



FLIP-IT II

Apple® Integer BASIC version by Michael Prescott
TRS-80® translation by Alan J. Zett

Flip-It II is a computerized version of Reversi for a TRS-80® Model I or III with 16K.

Flip-It II is a computerized board game in which you and the computer match wits to outflank and capture one another's pieces on an eight-by-eight board. The computer is a formidable opponent, and you won't find it a trivial matter to win.

The game begins with a square arrangement of four chips in the center of the board, two of them yours, and two belonging to the computer. You first choose a color and determine who will play first. You may also set up your own board (great for practicing strategies).

The game's object is to place one of your chips in an unoccupied square so that it outflanks one or more of the computer's chips, i.e. surround the computer chips with one of your existing chips and the new one you're playing, in a straight line. When you accomplish this, all the computer's outflanked pieces become yours. This can happen in more than one direction. In any given turn, you might capture pieces horizontally, vertically, and diagonally, resulting in a substantial shift in the relative number of chips in one turn. The outcome of the game is rarely certain until the last few moves.

To enter your move, type the number of the square you want. The computer always checks to see that your move is legal and doesn't allow cheating.

During your turn, you may ask the computer to recommend your best move by pressing "B". If you see no possible move, press "N"; the computer then checks to see if you really have no possible move. If so, it continues with its play. If the computer has no move, it passes, after searching all the open squares. If neither of you has a possible move (which occasionally happens even before the board is filled), the game is over and the player with the greater number of chips is the winner. You may also press "Q" to quit at any time.

You may type "R" to redraw the game board. This is useful if you accidentally halt execution of the game. Simply type the CONT command, then press "R". These also allow the "C" option to change the game board during play. The "C" option works just like the set-up game routine.

Winning a game involves more than capturing as many pieces as you can on any given turn. Much more important to the eventual result is the position of your chips on the board. Capturing edge and (especially) corner squares, and preventing the computer from doing the same, pays off in the long run, even if it means outflanking only one square when you could get more elsewhere on the board.

FLIP-IT II

Variables

A(*): Decision-making array for computer's moves.

A\$: General purpose (mainly INPUTs).

B(*): Array representing board layout. (Contents: 9 represents off-board, 0 is unoccupied square, 1 is occupied by white, -1 is occupied by black.)

B\$: Graphics string for a black piece.

BL: Black piece value in board array (= -1).

BP: Number of black pieces on board.

C: Value of the color (black = -1, white = 1) calling the "legal check" routine.

F: At start of game, F = 1 if you choose to go first. During game, used as a flag to flip pieces.

FL: If FL = 0, then legal check will not flip pieces even if a legal square was found.

IP: "I pass" flag for computer.

LG: Indicates legal move if = 1.

OP: Holds the value of the

computer's color (-1 or 1).

OP\$: Graphics string for the computer's pieces.

P: Temporary holding variables for PDL.

P2: Total number of pieces on the board.

PDL: Position of a square under consideration (1st digit is row, 2nd is column).

RF: Return flag; if = 1 then RETURN from a normal (non-GOSUB) routine.

SC: Value of the color being searched for in the "legal check" routine.

W\$: Graphics string for a white piece.

WH: White piece value for use in the board array (= 1).

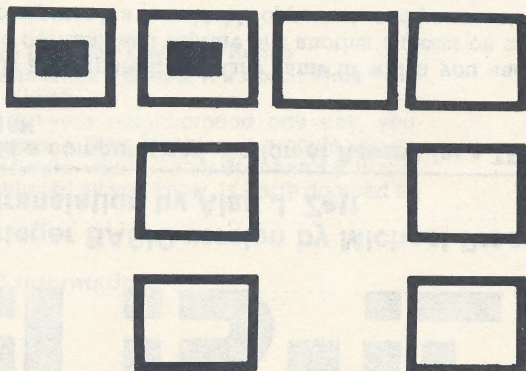
WP: Number of white pieces on the board.

X, Y, Z: Miscellaneous.

YO: Holds the value of a player's color (-1 or 1).

YO\$: Graphics string for your pieces.

YP: "You pass" flag.




```

SS SS SS SS SS SS SS SS SS SS SS
SS
SS TRS-80 BASIC SS
SS 'FLIP-IT II' SS
SS AUTHOR: MICHAEL PRESCOTT SS
SS TRANSL: ALAN J. ZETT SS
SS COPYRIGHT (C) 1983 SS
SS SOFTSIDE PUBLICATIONS, INC SS
SS
SS SS SS SS SS SS SS SS SS SS SS

```

If you don't wish to type this program, it is available on #39 SoftSide CV and DV.

Initialization.

```

10 CLS: CLEAR 200: DEFINITE Z: DIM A(64), B(99): A$ = "1181188831611383168
6386833633666415114841585485843533464356546564252247425754757326
22373267637672171128217872878227227744555445": Z = 1
20 FOR X = 1 TO 128 STEP 2: A(Z) = ASC(MID$(A$, X, 1)) - 48 + (ASC(MID$(A$, X + 1, 1)) - 48) * 10: Z = Z + 1: NEXT X: GOSUB 580: CLS

```

Initialize game board and graphic strings.

```

30 FOR X = 0 TO 99: B(X) = 0: NEXT X: B(45) = -1: B(54) = -1: B(44) = 1: B(55) = 1: BP = 2: WP = 2: P2 = 4
40 PDL = 0: B = 0: FL = 0: P2 = 0: RF = 0: BL = -1: WH = 1: OP = 0: YO = 0: FOR X = 0 TO 9: B(X) = 9: B(X + 90) = 9: NEXT X: FOR X = 9 TO 89 STEP 10: B(X) = 9: B(X + 1) = 9: NEXT X
50 B$ = CHR$(191) + STRING$(3, 131) + CHR$(191) + STRING$(5, 24) + CHR$(26) + STRING$(5, 131): W$ = STRING$(5, 191) + STRING$(5, 24) + CHR$(26) + STRING$(5, 131)

```

Get choice of color, who goes first, and whether or not a normal board is to be used. Also set up players' pieces.

```

60 CLS: F = 0: A$ = "": PRINT 24, "F L I P - I T": PRINT: PRINT: INPUT "WHICH COLOR DO YOU WANT (W OR B) "; A$: IF LEFT$(A$, 1) = "B" THEN B0
70 Y0$ = W$: OP$ = B$: Y0 = WH: OP = BL: GOTO 90
80 Y0$ = B$: OP$ = W$: Y0 = BL: OP = WH
90 A$ = "": PRINT: INPUT "DO YOU WANT TO GO FIRST (Y OR N) "; A$: IF LEFT$(A$, 1) <> "N" THEN F = 1
100 A$ = "": PRINT: INPUT "WANT TO SET UP YOUR OWN GAME (Y OR N) "; A$: CLS: GOSUB 110: IF LEFT$(A$, 1) = "Y" THEN B(44) = 0: B(45) = 0: B(54) = 0: B(55) = 0: WP = 0: BP = 0: P2 = 0: GOTO 210 ELSE IF F = 1 THEN 290 ELSE GOSUB 120: GOTO 360

```

Draw game board.

```

110 FOR X = 0 TO 7: F0RY = 0 TO 7: PRINT 24, X * 128 + Y * 7, STR$(X + 1); STR$(Y + 1);: NEXT Y: NEXT X: RETURN

```


Plot pieces and update game display data.

```

120 P2=0:WP=0:BP=0:FORX=0TO99:IFB(X)<>0ANDB(X)>9THENY=INT((X-11
)/10)*12B+((X-11)-INT((X-11)/10)*10)*7 ELSE 150
130 IFB(X)=1THENPRINT@Y,W$;WP=WP+1:P2=P2+1
140 IFB(X)=-1THENPRINT@Y,B$;BP=BP+1:P2=P2+1
150 NEXT:FORY=0TO47:SET(110,Y):NEXT:PRINT@56,"FLIP-IT";PRINT@18
5,"W =" ;WP;PRINT@249,"B =" ;BP;RETURN

```

No-possible-move command.

```

160 PRINT@376,"CHECKING";FORZ=1TO60:PDL=A(Z):P=PDL:C=YD:SC=OP:F
L=0:GOSUB460:IFLG=1THEN170ELSENEXT:IFIP=1THEN390ELSEYP=1:GOTO360
170 PRINT@376,"?WHAT ";PRINT@440,"ABOUT ";A$=LEFT$(STR$(PDL
),2)+ " "+RIGHT$(STR$(PDL),1):FORX=1TO15:PRINT@504,A$;FORY=0TO99
:NEXT:PRINT@504," ";:FORY=0TO99:NEXT:NEXT:GOTO300

```

Find player's best move.

```

180 PRINT@376,"LOOKING ";FORZ=1TO60:PDL=A(Z):P=PDL:C=YD:SC=OP:F
L=0:GOSUB460:IFLG=1THEN190ELSENEXT:IFIP=1THEN390ELSEYP=1:GOTO360
190 PRINT@440,"LOOKS ";PRINT@504,"GOOD!";A$=LEFT$(STR$(PDL),
2)+ " "+RIGHT$(STR$(PDL),1):FORX=1TO15:PRINT@376,A$;FORY=0TO99:N
EXT:PRINT@376," ";:FORY=0TO99:NEXT:NEXT:PRINT@504," ";
:GOTO300

```

Quit-game routine.

```

200 A$="":CLS:PRINT@24,"F L I P - I T":PRINT:INPUT"DO YOU REALLY
WANT TO QUIT (Y OR N) ";A$:IFLEFT$(A$,1)<>"Y"THENCLS:GOSUB110:G
OTO290ELSE400

```

Set up your own board.

```

210 RF=1:PRINT@504,"(D=DONE)";GOSUB300:IFA$="D"THEN230ELSEPRINT
@376,"COLOR? ";
220 A$=INKEY$:IFA$=""THEN220
230 IFA$="D"THENFORX=5TO7:PRINT@X*64+56," ";:NEXT:GOSUB11
0:P2=WH:WP=WH:BP=WH:IFF=1THEN290ELSEGOSUB120:GOTO360
240 PRINT@INT((PDL-11)/10)*12B+((PDL-11)-INT((PDL-11)/10)*10)*7,
;
250 IFA$="W"THENPRINTW$;B(PDL)=WH
260 IFA$="B"THENPRINTB$;B(PDL)=BL
270 IFA$=CHR$(31)THENPRINTSTR$(INT(PDL/10));" ";RIGHT$(STR$(PDL
),1);" ";STRING$(5,24);CHR$(26);" ";B(PDL)=0
280 GOTO210

```


Get player's move.

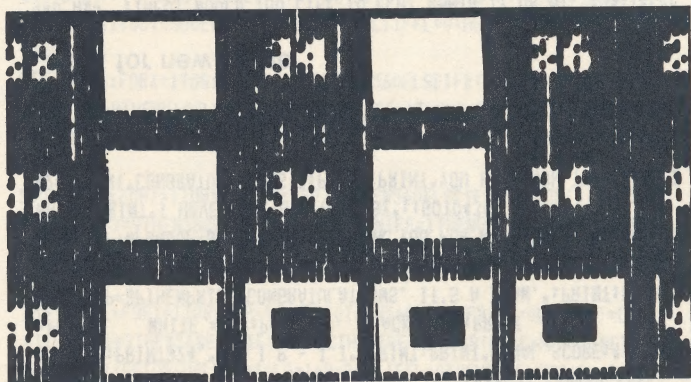
```
290 IFP2=64ORWP=0ORBP=0THEN390ELSEGOSUB120:IFP2=64ORWP=0ORBP=0TH
EN390
300 PDL=0:YP=0:PRINT@440,"          ";:PRINT@376,,:IFRF=1THENPRINT"
SQUARE? ";ELSEPRINT"MOVE? ";
310 FORX=0TO1
320 A$=INKEY$:IFA$=""THEN320ELSEIFA$="C"ANDRF=0THENF=1:GOTO210EL
SEIFA$="R"ANDRF=0THENCLS:GOSUB110:GOSUB120:GOTO300ELSEIFA$="Q"AN
DRF=0THEN200ELSEIFA$="N"ANDRF=0THEN160ELSEIFA$="B"ANDRF=0THEN180
ELSEIFA$="D"ANDRF=1THENRF=0:RETURN
330 IFA$<"1"ORA$>"8"THENPRINT@440,"INVALID";:FORY=1TO555:NEXT:GO
TO300
340 PRINT@441+X*2,A$;:PDL=PDL+VAL(A$)*10[(1-X):NEXT:IFRF=1THENRF
=0:RETURN
350 C=Y0:SC=0P:FL=1:GOSUB460:IFLG=0THENA$="0":GOTO330ELSEGOSUB12
0
```

Choose computer's move.

```
360 IFP2=64ORWP=0ORBP=0THEN390
370 IP=0:PRINT@376,"MAYBE ";:FORZ=1TO64:PDL=A(Z):PRINT@440,LEF
T$(STR$(PDL),2)" "RIGHT$(STR$(PDL),1);:P=PDL:IF B(P)<>0THENNEXTZ
ELSEC=0P:SC=Y0:FL=1:GOSUB460:IFLG=0THENNEXTZELSE290
```

Computer must pass.

```
380 IF P2<>64 AND YP=0 THEN PRINT@569,"PASS!";:IP=1:FORX=0TO555:
NEXTX:PRINT@569,"          ";:GOTO290
```



FLIP-IT II

End-game routine.

```

390 PRINT@568,"NO MOVES";:PRINT@634,"LEFT!";:PRINT@760,"# GAME #
";:PRINT@824,"# OVER #";:FORX=0TO5000:IFINKEY#(<>) "THENNEXTX
400 CLS:PRINT@24,"F L I P - I T":PRINT:PRINT"FINAL SCORE:":PRINT
:PRINT"    WHITE ="WP:PRINT"    BLACK ="BP:PRINT
410 IFWP=BPTHENPRINT"CONGRATULATIONS. IT'S A DRAW.":PRINT:PRINT"
WE APPEAR TO BE EVENLY MATCHED.":PRINT:GOTO440ELSEIF(WP>BPANDOP=
WH)OR(BP>WPANDOP=BL)THENPRINT"THANK YOU FOR A STIMULATING GAME":
PRINT:PRINT"I HAVE WON THIS GAME BY";:GOTO430
420 PRINT"CONGRATULATIONS!":PRINT:PRINT"YOU HAVE WON THIS GAME B
Y";
430 PRINTABS(WP-BP)"CHIPS.":PRINT

```

Check for new game.

```

440 A$="":INPUT"WOULD YOU LIKE TO PLAY AGAIN (Y OR N) ";A$:IFLEF
T$(A$,1)<>"N"THENCLS:GOTO 30
450 CLEAR50:PRINT:PRINT"GOOD BYE...":END

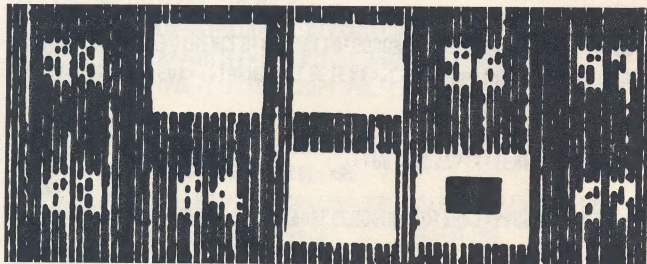
```

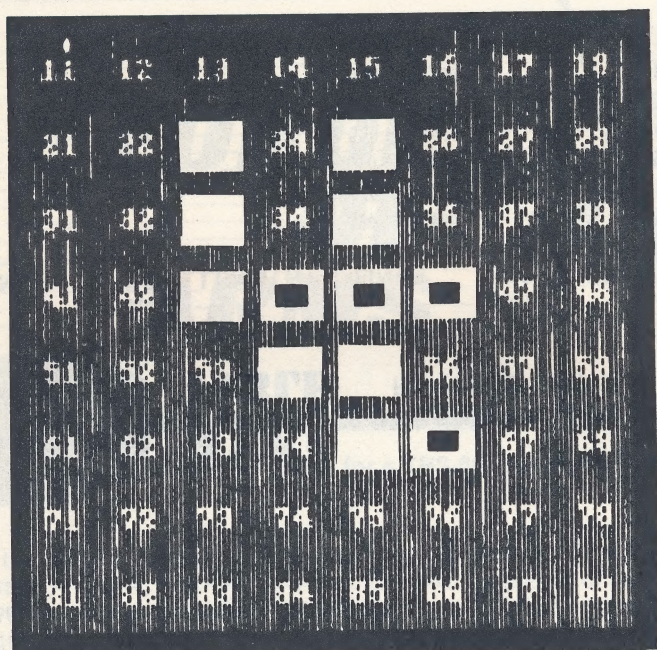
Legal check routine. Combines a search for legal moves with a routine to flip pieces.

```

460 LG=0::IFB(PDL)<>0THEN570
470 FORY=1TO8:F=0:ONYGOTO480,490,500,510,520,530,540,550
480 P=PDL:FORX=1TO8:P=P+10:IFP>99THEN560ELSEIFB(P)=0OR(B(P)=CAND
X=1)ORB(P)=9THEN560ELSEIFB(P)=SCTHENIFF=1THENB(P)=C:NEXTX:GOTO56
0ELSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOT
O480
490 P=PDL:FORX=1TO8:P=P-10:IFP<0THEN560ELSEIFB(P)=0OR(B(P)=CANDX
=1)ORB(P)=9THEN560ELSEIFB(P)=SCTHENIFF=1THENB(P)=C:NEXTX:GOTO560
ELSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOTO
490
500 P=PDL:FORX=1TO8:P=P+1:IFP>99THEN560ELSEIFB(P)=0OR(B(P)=CANDX
=1)ORB(P)=9THEN560ELSEIFB(P)=SCTHENIFF=1THENB(P)=C:NEXTX:GOTO560
ELSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOTO
500

```





```

510 P=PDL:FORX=1TO8:P=P-1:IFP<0THEN560ELSEIFB(P)=0OR(B(P)=CANDX=
1)ORB(P)=9THEN560ELSEIFB(P)=5CTHENIFF=1THENB(P)=C:NEXTX:GOTO560E
LSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOTO5
10
520 P=PDL:FORX=1TO8:P=P+9:IFP>99THEN560ELSEIFB(P)=0OR(B(P)=CAND
X=1)ORB(P)=9THEN560ELSEIFB(P)=5CTHENIFF=1THENB(P)=C:NEXTX:GOTO56
0ELSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOT
O520
530 P=PDL:FORX=1TO8:P=P-9:IFP<0THEN560ELSEIFB(P)=0OR(B(P)=CANDX=
1)ORB(P)=9THEN560ELSEIFB(P)=5CTHENIFF=1THENB(P)=C:NEXTX:GOTO560E
LSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOTO5
30
540 P=PDL:FORX=1TO8:P=P+11:IFP>99THEN560ELSEIFB(P)=0OR(B(P)=CAND
X=1)ORB(P)=9THEN560ELSEIFB(P)=5CTHENIFF=1THENB(P)=C:NEXTX:GOTO56
0ELSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOT
O540
550 P=PDL:FORX=1TO8:P=P-11:IFP<0THEN560ELSEIFB(P)=0OR(B(P)=CANDX
=1)ORB(P)=9THEN560ELSEIFB(P)=5CTHENIFF=1THENB(P)=C:NEXTX:GOTO560
ELSENEXTX:GOTO560ELSEF=1:LG=1:P=PDL:IFFL=0THEN570ELSEB(P)=C:GOTO
550
560 FORX=0TO1:NEXTX:NEXTY
570 FORX=0TO1:NEXTX:FORY=0TO1:NEXTY:RETURN

```


Game instructions.

```

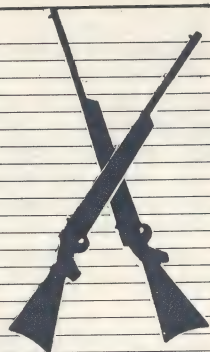
580 A$="":PRINT@512,,:INPUT"NEED INSTRUCTIONS (Y OR N) ";A$;IFLE
FT$(A$,1)<>"Y"THENRETURN
590 CLS:PRINT@24,"F L I P - I T":PRINT:PRINT"THE OBJECT OF THIS
GAME IS TO COMPLETELY FILL THE BOARD WITH AS MANY PIECES OF YOUR
COLOR AS YOU CAN. TO DO THIS YOU MUST OUT- FLANK YOUR OPPONENT
'S PIECES AND FLIP THEM TO YOUR COLOR."
600 PRINT:PRINT"OUTFLANKING CAN OCCUR HORIZONTALLY, VERTICALLY,
OR DIAGONALLY. THE GAME ENDS WHEN THE BOARD IS FULL, OR WHEN BO
TH PLAYERS CAN'TMOVE. AT THAT TIME, WHOEVER HAS THE MOST PIECES
WINS."
610 PRINT@977,"(PRESS <ENTER> TO CONTINUE)";
620 IFINKEY$(<>CHR$(13))THEN@620
630 CLS:PRINT@24,"F L I P - I T":PRINT:PRINT"YOU MAKE MOVES BY E
NTERING THE ROW NUMBER COMBINED WITH THE COLUMN NUMBER. FOR
EXAMPLE: THE UPPER RIGHT-HAND CORNER IS THE POSITION 18 - THAT
IS 1 FOR THE FIRST ROW AND 8 FOR THE EIGHTH"
640 PRINT"COLUMN. IF YOU MAKE A MISTAKE ENTERING A NUMBER, TYPE
A NINE FORANY OF THE NUMBERS AND IT WILL MARK THE ENTRY AS INVAL
ID.":PRINT"DURING THE GAME YOU ALSO HAVE THE FOLLOWING OPTIONS:"
:PRINT
650 PRINT"Q - TO QUIT THE GAME":PRINT"R - REDRAW THE GAME BOARD"
:PRINT"N - NO AVAILABLE MOVE":PRINT"C - CHANGE GAME BOARD SETUP"
:PRINT"B - ASK FOR THE BEST MOVE":PRINT@977,"(PRESS <ENTER> TO S
TART GAME)";
660 IFINKEY$(<>CHR$(13))THEN@660ELSERETURN

```



LINES	SWAT CODE	LENGTH	LINES	SWAT CODE	LENGTH
10 -	50	UX	504	380 -	430
60 -	120	QG	532	52	520
130 -	190	DJ	625	440 -	500
200 -	290	CQ	531	CG	611
300 -	370	ET	547	510 -	540
				TE	576
				550 -	600
				YG	664
				610 -	650
				SG	650
				660 -	660
				BF	21

A FORCE OF ARMS



by Steven J. Choi

Force of Arms is a strategy game for a TRS-80® Model I or Model III with 32K RAM.

Only the craftiest general can overcome his adversaries in this war of cunning, strategy, chance, and *Force of Arms*. Can you conquer all of Earth's territories, and dominate the globe? Then take command of your armies! Defend the home front, and march forward in your quest for conquest.

Force of Arms is centered around a graphic map of the world (three screens long by two screens wide), divided into 42 territories. At the start, the game will ask you if you wish to distribute the territories, or prefer the computer to distribute them randomly. If you decide to distribute territories, you will, in turn, be asked the number of the territory you wish to control. During this phase you use the four arrow keys to display different sections of the map. To select a territory, type its number. You need not hit ENTER afterwards.

At the beginning of each turn, reinforcements are calculated and awarded to the current player. Then the game gives the player a set of options with which to reinforce territories, attack other players, check game status, or pass his turn.

Options

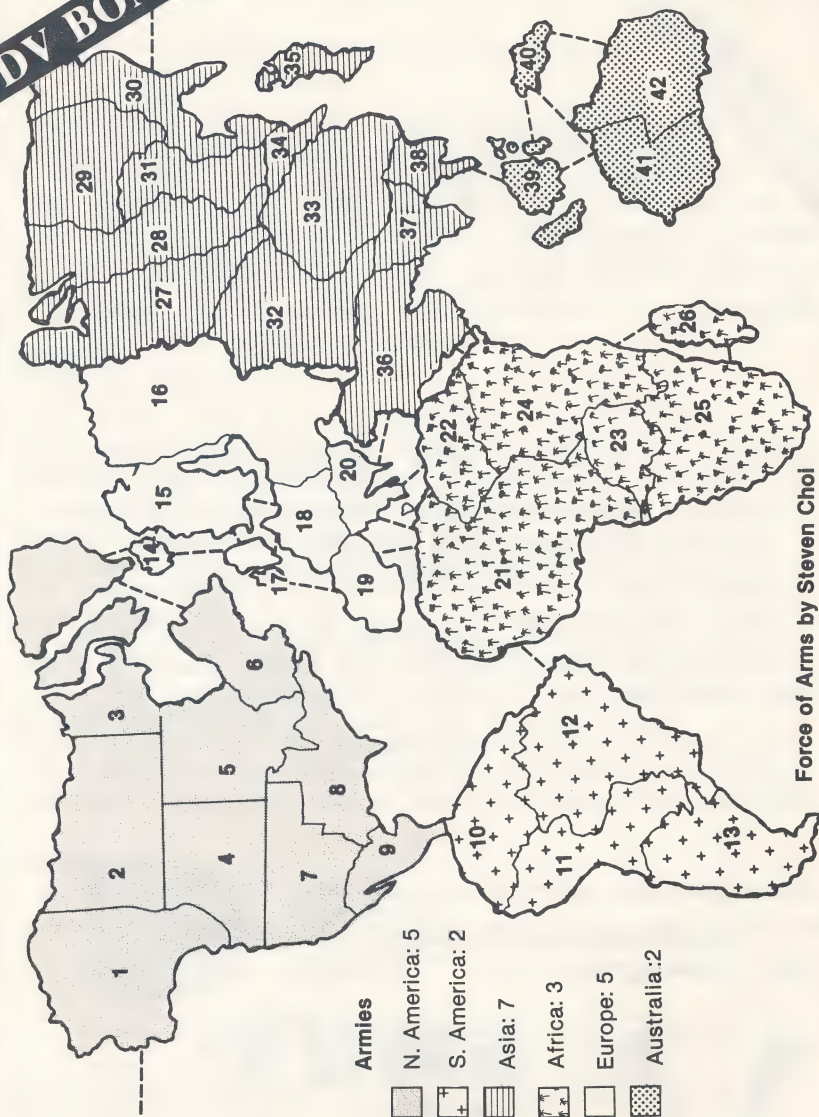
- The **Reinforcement Command** is command option one. The number of reinforcements you receive depends on the number of territories you occupy, the number of continents you totally occupy, and the number of bonuses you hold. For every three territories, you add one army to your reinforcements (with a minimum of three armies added per turn). For every continent totally occupied, a certain number of armies are added to your reinforcements, depending on which continent you control. You receive one bonus every turn — provided you conquer at least one territory.

DV BONUS

— **FORCE OF ARMS** —

FORCE OF ARMS

DV BONUS



Force of Arms by Steven Choi

At the beginning of each turn, the player receives bonus reinforcements if he has three, four, or five bonuses. The chance of receiving the bonus reinforcements rises in proportion to the number of bonuses you hold. The number of bonus reinforcements you receive increases for every bonus you turn in. For instance, for the first bonus, two armies are given, for the second bonus, four armies, for the third, six armies, and so on. To allocate reinforcements to a territory, you must enter the territory's name or number, and the number of armies to send in.

● The **Player Report Command** is command option two. This command calls up the player report routine, which displays information on what territories are occupied by each player.

● The **Continent Status Command** is command option three. This command asks you which continent to examine and prints out the status of each territory (number of armies, who occupies, and so on) in that continent.

● The **Territory Report Command** is command option four. For this command, you receive a list of all 42 territories and their computer numbers. Enter the number of the territory you wish to examine, and the computer prints out a report showing who occupies it, how many armies they have, and who borders the territory.

● The **Attack Command** is command option five. Use this command to attack an opponent's territory from one of yours. To attack a territory, the attacker must have at least two armies in the territory from which the attack is launched, and your attacking territory must be connected to the defending territory — by a land or water route. In an attack, the defender has a slight advantage over the attacker, meaning that with even odds, the defender will usually win.

● The **Pass Command** is command option six. You use this command when you feel you have done all you can during a turn and wish to end it. The player ending a turn has a last chance to move armies from one territory to an adjacent territory (by water or land). To make a last move from a territory, the territory must have more than one army, and at least one army must remain after you make the move.

● You can call the **Help Routine** by pressing "H". The routine will print out a summary of these instructions.

● You can call the **Save Game Feature** by pressing "S". Respond to the prompt with a legal filename to save your game. Instead of a filename, you may type "END" to terminate the game without saving it. If you specify a legal filename, the computer will ask if you wish to continue the game.

The rest of the commands control the graphic map. With the arrow keys, you can move among six possible **Map Configurations**. The clear key switch toggles between the two **Map Options**. Option one, *General Map*, shows the graphic map with the computer number of each territory within the corresponding territory. Option two, *Strategic Map*, shows the graphic map with a two character status report within each territory. The first character shows who occupies the territory; the second shows how many armies he has positioned there.

Playing Notes

You may identify a territory for the computer with either its number or name. Since the graphic map is divided into six sections, you will be unable to see some territory boundaries (for example, Ukraine to Ural). To overcome this, consult your player's map which shows all territory boundaries and territory numbers.

Generally, you should try to capture a continent (for the bonus armies), and keep your own territories adjacent to each other. This will lower the number of fronts you must protect, and allows you to maneuver armies from your protected territories to the front line.

Savo Island

by Victor A. Vernon, Jr.



Savo Island is a battle simulation game for the TRS-80® I and III with 16K RAM.

The Battle of Savo Island is an obscure WW II skirmish which the U.S. Navy would like us to forget. After the defeat of the Japanese carrier force at Midway, the Pacific War went into a two-month lull. The only action of any kind was the steady, but slow, movement of the Japanese over the Owen Stanley Mountains towards Port Moresby, on the Island of New Guinea.

During the lull, a strategic controversy confronted the Allies. Both Adm. King and Gen. MacArthur proposed an offensive against the Japanese in the Bismark-New Guinea area. MacArthur wanted a direct assault on the Japanese base at Rabaul. MacArthur would command this assault, with Navy air support from its carriers.

The Navy wanted no part of this plan for two reasons. Foremost, they wanted their carriers and support ships out of range of the land-based aircraft on Rabaul. Secondly, the idea of an Army General commanding the Pacific Fleet was galling to proud Navy personnel.

Adm. King wanted to "island hop" up the Solomon Islands toward Rabaul while the Allies established land bases to support further advances. MacArthur felt this plan would deprive him of the troops he needed to defend Port Moresby.

The Joint Chiefs settled this clash of personalities and service jealousies with a compromise, three-phase operation which included:

- Seizure of the northeast coast of New Guinea by Gen. MacArthur.
- Seizure of the Southern Solomons by Vice Adm. Robert L. Ghormley.
- A combined operation against Rabaul later. The command structure of the third phase would be determined when it was initiated.

The attack on Rabaul never materialized. After MacArthur secured the northeast New Guinea Coast, the Navy built a ring of strategic bases around Rabaul, making Rabaul useless to the Japanese from late 1943 to the end of the war.

MacArthur then turned his attention to advancing along the northern New Guinea Coast towards the Phillipines, while the Navy advanced through the central Pacific, towards Japan. The problem of who commanded what would reappear before the Leyte Operations and during planning for "Operation Olympic."

The fact that the Japanese were building an airfield on Guadalcanal was not the sole reason for the invasion of Guadalcanal, as most history books claim. However, the airfield did make Guadalcanal the primary target and sped up the timetable for the operation.



The Battle Begins

Early on the morning of August 7, 1942, exactly eight months after the attack on Pearl Harbor, an Allied amphibious force, consisting of four U.S. cruisers, three Australian cruisers, nineteen U.S. destroyers, and nineteen troop transports, landed the U.S. First Marine Division on the north shore of Guadalcanal. More Marines landed on Tulagi Island. The first step on the long road to Tokyo had been taken. Surprise was complete. Within 24 hours, all objectives had been taken against token opposition. Air support was supplied by Adm. Fletcher's three-carrier task force.

The Japanese were slow to react on the beach, but they struck back quickly in the air. Adm. Inouye at Rabaul launched three separate air attacks, and while they inflicted little actual damage, they did interfere with the landings. Meanwhile, a Naval task force of seven cruisers and one destroyer, under the command of Vice Adm. Gunichi Mikawa, sailed south through "The Slot" towards Guadalcanal. This is the force our program will simulate.

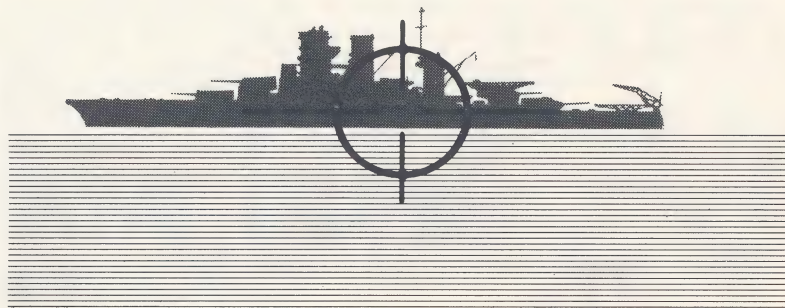
The following day, Japanese planes continued their intense air attacks in a series of violent air battles staged over the sound between Guadalcanal and Tulagi. After this, Tulagi would be called Iron-Bottom Sound, because the hulls of dead ships would cover its floor. Japanese planes also attacked the U.S. carriers so Adm. Fletcher decided to withdraw them, leaving Adm. Turner to command the landing and support forces, without air cover. Turner decided to retreat also, but delayed his departure until August 9, 1942. Unfortunately for Turner, Adm. Mikawa arrived shortly after midnight on the very day Turner planned to leave.

Mikawa's force, passing south of Savo Island into Iron-Bottom Sound, surprised Turner's patrolling cruisers. In 32 minutes of battle, characterized by excellent Japanese gunnery and ship handling, four of five Allied, heavy cruisers (one American & three Australian) were sunk, and the fifth damaged.

Mikawa, fearing American air attack at dawn, broke off the engagement and headed back to Rabaul. So complete and so fast was his victory that the Japanese ships were virtually untouched. Only 37 men were killed and 57 wounded. Mikawa did not know Fletcher had withdrawn.

Of the two U.S. destroyers guarding the entrance to the sound, one, the U.S.S. Blue, knew nothing of the battle until the next day. The other, the U.S.S. Talbot, discovered the Japanese as they were leaving the battle area.

SAVO ISLAND



Outnumbered eight to one, she survived only by ducking into a rain squall.

Thus ended the Battle of Savo Island. Even though it was an Allied defeat, it, and the Guadalcanal landings, marked the turning point of the Pacific War. Forget what they told you about Midway. The turning point was the entire campaign for Guadalcanal. After the war, when Adm. Kurita was asked what he thought was the turning point, without hesitation he answered "Guadalcanal!"

The Savo Island simulation puts you in the unenviable position of trying to change history. You will command the Allied ships. The computer plays Adm. Mikawa.

Unlike some battle simulations, this one is designed to be as faithful to history as possible. Because of this, several restrictions will hamper you. These restrictions make this a very challenging simulation.

Now, for the problems you will encounter:

1 — Since this was a night action, the Japanese ships will *not* be displayed on the screen unless (A) they shoot at you or (B) one or more of your ships are close enough to "spot" them.

2 — When a Japanese ship does appear, it will look exactly like your ships. Remember, at night, one ship looks like another.

3 — At the start of the battle, a Japanese float plane dropped flares to the east of the battle area. This provided the Japanese very good silhouetting of the Allied ships. The Japanese also had better night binoculars. This means that the Japanese (computer) gunfire will be devastatingly accurate. Your gunfire will be only as accurate as you are. I will explain the gunfire routine later.

4 — During the whole war, and especially early in the war, a very unreliable torpedo, called the "Mark 14," hampered the U.S. Navy. The Navy tried various warheads, from contact to proximity (magnetic). The torpedo often failed to run at the prescribed depth, which sometimes caused the thing to breach (come out of the water) and explode against a wave, or run too deep and sink. At other times, the torpedo would hit the side of a ship and go "thunk." Nothing more. The Japanese were not plagued with this problem. Their "Long Lance" torpedo was one of the best in the war. As you can probably guess, you will have the same problem.

5 — When you fire guns or torpedoes, the computer will compute where your shot(s) will hit. If you hit one of your own ships, the shell will do as much damage to your ship as it would to a Japanese ship. Remember: "The ship you sink may be your own!"

Those are the general limitations and characteristics of the simulation.

How To Play

The screen display is divided into two parts. The upper part is a map of the battle area. North is the top of the screen; East is the right side. South is the bottom and West is the left. The map is distinguished by three land masses. Tulagi Island is located on the northeast corner, and Guadalcanal is directly south of Tulagi on the southeast edge of the map. Savo Island is in the center. Your ships will be placed "on station," one at a time, and you will then learn their names.

The lower half of the screen is where all the messages to you will be displayed. Consider this your "work area." You will see immediately that you have four "commands" to choose from. They are:

- Change Course
- Change Status
- Fire Weapons
- No Action

The length of time you have to decide what to do is limited. The time remaining is displayed on the right side of the message area, directly below Guadalcanal.

The overall flow of the game is:

- 1 — Allied Attack Phase (Optional)
- 2 — Japanese Movement
- 3 — Allied Movement
- 4 — Japanese Attack Phase

These are the basic parts. You can change course or status at almost any time as long as you do it before going to the Allied Attack Phase (Fire Weapons). If you take "No Action" the computer will go to the Movement Phases.

Movement is automatic on a predetermined course. You can change the course of your ships when you select the "Change Course" command. When you do, the present course of each ship will be displayed, along with your options. To keep a ship on the same course, enter its present course. Two "don'ts" to remember: If two ships collide, it will do more damage than a torpedo hit. If one of your ships runs aground or leaves the battle area, it is considered *Sunk*.

The second selection, "Status Change," will enable you to change the speed and/or status of your ships, with one restriction. If none of the Japanese ships are spotted, you can only change the status of one ship. At the start, all of your ships, except the picket destroyers, are at full speed and at Condition II. The pickets are at flank speed and General Quarters. Your three speed options are: Standard, Full, & Flank. Your three status options are: Normal (actually Condition III, Wartime Non-combat), Condition II (Half the crew on watch), and General Quarters (Battle Stations).

Firing the guns is a little complicated so I will go through it step-by-step. You have two gun sizes on the Cruisers: main eight-inch and secondary four-inch. While it is true that some of the Allied Ships had five-inch and not four-inch guns, I decided to lump them all into the secondary category, and call it four-inch. The Destroyers only have the smaller armament, but carry torpedoes. When you fire the guns, you will be asked for two things, a Range and a Bearing. The Range of the eight-inch guns is between two and 20. A four-inch gun's Range is between two and ten. If you do not wish to fire a ship's guns, enter zero for a Range.

SAVO ISLAND

The Bearing of the shot is anything in the arc of a complete circle with the ship in the center of the circle. North is, of course, both zero degrees and 360 degrees. You can enter anything between zero and 360, including fractions of a degree. However, I have found that fractions are not needed. As an aid, remember 90 degrees is due east, 180 degrees is due south, and 270 degrees is due west.

After you have entered the Range and Bearing, the computer will fire a "spread" of shots determined by how many guns you are firing. For example, suppose you wish to fire the eight-inch guns of the U.S.S. Chicago. The computer will note nine big guns on the Chicago. It will then subtract nine from the Bearing you have entered. If you entered 90, then your first shot will be at 81 degrees (90 minus nine). It will then fire eight more shots, and will add two on each shot. Thus, your shots will be at 81, 83, 85, 87, 89, 91, 93, 95, and 97 degrees. Note that the initial entry of 90 degrees is not used in this example. However, because of the computations, the figures of 89 and 91 will put the shot at essentially the same place. This only occurs with ships with odd numbers of guns. As you can see, the original Bearing of 90 degrees is in the approximate center of the spread.

As each shot is fired, it will be displayed on the screen as a "set point," unless it is a direct hit on a ship or on land. You will then be told if you hit anything. You can also "straddle" a ship by landing a shot directly above, below or to either side of it. While straddling a ship will not do as much damage as a direct hit, it will do some damage.

Three things to remember:

1 — If you sink a ship on one shot, then hit it again with another shot, the second shot will not be recorded as a hit. (You can't hit a sunken ship. The ship remains visible because it is assumed that all shells are fired simultaneously. Only the report to you is done sequentially.)

2 — Under the right conditions, you can straddle the firing ship. I included this unintentionally, and decided to leave it to simulate "short" shots and other combat accidents.

3 — Because the Japanese ships are only visible during the movement phase (after their firing phase), it is possible for them to fire at you before you can see them.

The Range factor is actually a number of "points" on the TRS-80 screen. Because of the non-linear 4:3 ratio of the monitor screen, Tandy made each screen location two wide and three high. Therefore, if you enter a bearing between 315 degrees and 45 degrees or between 135 degrees and 225 degrees, the Range you enter will decrease to 2/3.

Firing torpedoes is essentially the same, except the Range factor is unnecessary, as torpedoes keep going until they hit a target or sink.

That is all you will need to know to play this simulation. You will need a lot more to "win." Do not expect to do so the first time you try, unless your name is Halsey. I wrote it and it took me 25 tries before I was able to sink all the Japanese ships. I still win only twenty percent of the time. If I made it too easy, it wouldn't be any fun.

To win, you must study the map, develop a sound strategy and stay with it. For more information on Savo, I suggest reading *The Campaign For Guadalcanal* by Jack Coggins, published by Doubleday, Inc.

Variables

A(n,nn): Holds all information on Allied Ships.

J(n,nn): Holds all information on Japanese Ships.

AS(n): Holds names of Allied Ships.

JS(n): Holds names of Japanese Ships.

A: Angle or bearing of Allied shot (used in Allied fire routine).

A1: Selects quadrant of Allied shot.

A2: Corrected angle for calculations.

A3: Horizontal screen coordinate of shot.

AS: Allied subtotal score.

AT: Allied total score.

A\$: CHR\$(31): clears bottom of screen.

B\$: Name of ship for status change routine.

B(n): Temporary storage of spotted flag (JAP).

BB: Flag for status change routine.

C\$: INKEY\$ variable.

C: Command input variable.

CO: Coefficient to change radians to degrees.

D: Random number determines if hit is scored by Japanese ships.

DD: Flag for set/reset Allied shots.

FL: Flag determines if any Allied ships afloat.

H: Used three ways.

1 — Hit factor, Allied surface fire.

2 — Flag for Allied torpedo fire.

3 — Horizontal coordinate when moving Allied ships.

H1: Hit factor, Allied torpedo fire.

J: Horizontal coordinate, Japanese movement.

JF: Japanese Flag, shows number of Japanese ships afloat.

JS: Japanese subtotal score.

JT: Japanese total score.

M: Vertical coordinate, Japanese Movement.

01: Together with A1, selects proper quadrant of Allied shot.

03: Vertical screen coordinate of Allied shot.

R: Range of Allied shot.

S: Loop counter.



SS: Loop counter.

S1: Caliber of gun shooting (Allied).

S2: Timing loop.

V: Vertical coordinate, Allied movement.

X: Loop counter.

X1: Actual horizontal location of Allied shot.

Y1: Actual vertical location of Allied shot.

Y: Used twice:

1 — Loop counter.

2 — Used in total score computation.

Z: Used twice:

1 — Loop counter.

2 — Torpedo hit flag.

The following Variables were added after the list of variables was made. C\$(n) was added to assist the player. G(N) was added to prevent "holes" on land if any Allied ships run aground.

C\$(n): Used to name off present course of Allied ships.

G(n): Flag for each Allied ship; shows whether ship has run aground or not.

SAVO ISLAND

```

SS SS SS SS SS SS SS SS SS SS SS
SS
SS TRS-80 BASIC SS
SS 'Savo Island' SS
SS Author: Victor Vernon Jr. SS
SS Copyright (c) 1983 SS
SS SoftSide Publications, Inc SS
SS
SS SS SS SS SS SS SS SS SS SS SS

```

If you don't wish to type this program, it is also available on SoftSide #39 CV and DV.

Needed for old TRS-80 ROMs (READ/DATA cure).

```
10 POKE16553,255
```

Title section.

```
40 GOTO450
```

Allied fire routine.

```

50 DD=0: IFH=3THEN70ELSEFORS=1TDA(X,S1):A=A+2: IFA>360THENA=A-360
60 PRINT@640,A$"SHOT #";A$
70 IFA=>270THENA2=A-270:O1=1:A1=-1
80 IFA=180THENA2=270-A:O1=1:A1=-1ELSEIFA=90THENA2=A-90:O1=1:A1=1
90 IFA=<90THENA2=90-A:O1=-1:A1=1
100 IFH=3THENH1=RND(3)+1ELSEH1=H+RND(2)
110 A3=(COS(A2*CO)) *R*A1:A3=FIX(A3):O3=(SIN(A2*CO)) *R*O1:O3=FIX(O3)
120 X1=A3+A(X,6):Y1=O3+A(X,7): IFX1<0ORX1>127ORY1<0ORY1>27THEN260
130 IFPOINT(X1,Y1)=0THENSET(X1,Y1):DD=-1
140 FORSS=0TO16: IFA(SS,3)<=0THEN190
150 IFA(SS,6)=X1ANDA(SS,7)=Y1PRINT@705,A$"DIRECT HIT ON "A$(SS):
A(SS,3)=A(SS,3)-H1:FORS2=1TO500:NEXTS2: IFH=3THENR=30
160 IFDDANDH=3THEN190ELSEIFDD=0ANDH=3THENR=30:GOTO190
170 IFA(SS,6)=X1AND(A(SS,7)=Y1+1ORA(SS,7)=Y1-1)PRINT@705,A$"STRATTLED "A$(SS):A(SS,3)=A(SS,3)-H1/2:FORS2=1TO150:NEXTS2
180 IFA(SS,7)=Y1AND(A(SS,6)=X1+1ORA(SS,6)=X1-1)PRINT@705,A$"STRATTLED "A$(SS):A(SS,3)=A(SS,3)-H1/2:FORS2=1TO150:NEXTS2
190 IFSS>7THEN250
200 IFJ(SS,3)<=0THEN250
210 IFJ(SS,4)=X1ANDJ(SS,5)=Y1PRINT@705,A$"DIRECT HIT ON "J$(SS):J(SS,3)=J(SS,3)-H1:J(SS,6)=-1:FORS2=1TO500:NEXTS2: IFH=3THENR=30
220 IFH=3THEN250
230 IFJ(SS,4)=X1AND(J(SS,5)=Y1+1ORJ(SS,5)=Y1-1)PRINT@705,A$"STRATTLED "J$(SS):J(SS,3)=J(SS,3)-H1/2:J(SS,6)=-1:FORS2=1TO100:NEXTS2
2

```

```

240 IFJ(SS,5)=Y1AND(J(SS,4)=X1+1ORJ(SS,4)=X1-1)PRINT@705,A$*STRA
TTLED "J$(SS):J$(SS,3)=J$(SS,3)-H1/2:J$(SS,6)=-1:FOR$2=1TO100:NEXT$
2
250 NEXTSS:IFDDTHENRESET(X1,Y1):DD=0
260 IFH<3THENNEXTS:RETURNELSERETURN

```

Course correction.

```

270 X=X-X/3:FORZ=0TO16:IFA(Z,3)<=0THEN320
280 PRINT@640,A$*PRESENT COURSE OF "A$(Z)" IS "C$(A(Z,4))
290 PRINT"ENTER NEW COURSE, 1 IS NORTH, 2 = N.E., 3 = EAST, ":
300 PRINT"4 = S.E., 5 = SOUTH, 6 = S.W., 7 = WEST, 8 = N.W."
310 C$=INKEY$:IFC$=""THEN310ELSEA(Z,4)=VAL(C$):IFA(Z,4)<1ORA(Z,4
)>8THEN310
320 NEXTZ:RETURN

```

Status and speed changes.

```

330 X=X-X/3:FORZ=0TO7:IFJ(Z,6)THEN440ELSENEXTZ
340 IFBBPRINT@640,A$*NO MORE CHANGES ALLOWED UNTIL AFTER MOVEMEN
T PHASE":FORZ=1TO800:NEXTZ:PRINT@640,A$:RETURN
350 PRINT@640,A$*YOU HAVE NOT SPOTTED ANY JAPANESE SHIPS THEREFO
RE YOU CAN ONLY CHANGE SPEED OR STATUS ON ONE SHIP.":BB=-1
360 INPUT"ENTER THE NAME OF THE SHIP YOU WISH TO CHANGE":B$
370 FORZ=0TO16:IFA$(Z)=B$PRINT@640,A$*ENTER NEW SPEED FOR "A$(Z)
ELSE430
380 PRINT"1 - STANDARD 2 - FULL SPEED 3 - FLANK SPEED";
390 C$=INKEY$:IFC$=""THEN390ELSEA(Z,5)=1.5*(VAL(C$)):IFA(Z,5)<1O
RA(Z,5)>5THEN390
400 PRINT@640,A$*ENTER NEW STATUS FOR "A$(Z)
410 PRINT"1 - NORMAL 2 - CONDITION II 3 - GENERAL QUARTERS";
420 C$=INKEY$:IFC$=""THEN420ELSEA(Z,9)=VAL(C$):IFA(Z,9)<1ORA(Z,9
)>3THEN420
430 NEXTZ:PRINT@640,A$*:RETURN
440 FORZ=0TO16:IFA(Z,3)<=0THEN430ELSEPRINT@640,A$*ENTER NEW SPEE
D FOR "A$(Z):GOTO380

```

Initialize program and arrays.

```

450 CLS:CLEAR100:DIMA(16,9),B(7),G(16),J(7,7),A$(16),J$(7),C$(8)
:CO=.0174533:A$=CHR$(31):DEFINTS-Z:RANDOM
460 PRINTCHR$(23)"THE BATTLE SIR,":PRINT"IS NOT TO THE STRONG AL
ONE,":PRINT"IT IS TO THE VIGILANT,":PRINT"THE ACTIVE, THE BRAVE.
":PRINTTAB(10)"- PATRICK HENRY -"
470 FORX=1TO8:READC$(X):NEXTX
480 FORX=0TO16:READA$(X):FORY=0TO9:READA(X,Y):NEXTY,X
490 FORX=0TO7:READJ$(X):FORY=0TO7:READJ(X,Y):NEXTY,X

```

Draw map and print message.

```

500 CLS:PRINT@40,CHR$(139);STRING$(23,191);:PRINT@106,STRING$(3,
131);STRING$(3,191);STRING$(4,131);CHR$(32)STRING$(2,191)STRING$
(3,143)STRING$(6,191);:PRINT@188,CHR$(131);STRING$(3,143);

```


SAVO ISLAND

```

510 PRINT@278,STRING$(5,191);:Y=11:FORX=46TO50:SET(X,Y):SET(X,Y+
5):NEXTX:RESET(50,Y):SET(43,14):FORX=43TO52:SET(X,15):NEXTX:PRIN
T@591,STRING$(16,191);STRING$(10,188)STRING$(23,191);
520 PRINT@528,STRING$(8,191);STRING$(3,176);:PRINT@557,STRING$(8
,188)STRING$(11,191);:PRINT@508,STRING$(4,188);:PRINT@640,A$;
530 PRINT"I WILL NOW LOCATE ALL YOUR SHIPS. PLEASE TAKE NOTE WHE
RE THEY ARE. TO SIMULATE THE CONFUSION OF BATTLE THIS WILL BE
THE LAST TIME YOU WILL BE TOLD WHO IS WHERE.":PRINT" PRESS AN
Y KEY TO CONTINUE.";
540 C$=INKEY$:IFC$=""THEN540

```

Put Allied ships on map, and name them.

```

550 FORZ=0TO16:PRINT@640,A$"NOW LOCATING THE "A$(Z);:FORX=1TO150
:RESET(A(Z,6),A(Z,7)):SET(A(Z,6),A(Z,7)):NEXTX,Z

```

Command input with time limit.

```

560 PRINT@640,A$;:FORX=100TO00STEP-5:PRINT@640,"PLEASE SELECT YO
UR COMMAND":PRINT"1 - CHANGE COURSE":PRINT"2 - CHANGE ST
ATUS":PRINT"3 - FIRE WEAPONS":PRINT"4 - NO ACTION";
570 PRINT@684,"TIME REMAINING"INT(X/100)+1;
580 C$=INKEY$:C=VAL(C$):IFC<1DRC>4THEN610
590 IFC=4THEN830
600 IFC=360TO620ELSEDCBGSUB270,330
610 NEXTX:60TO830
620 FORX=0TO16:IFA(X,9)<3DRA(X,3)<=0THEN760

```

Allied gunfire.

```

630 PRINT@640,A$"ONLY SHIPS AT GENERAL QUARTERS CAN FIRE"
640 IFX>7THEN710
650 PRINT@704,A$"ENTER BEARING IN DEGREES 0-360 FOR 8-INCH GUNFI
RE FROM "A$(X):INPUTA
660 IFA<0THENA=A+360:GOTO660ELSEIFA>360A=A-360:GOTO660
670 A=A-A(X,0):IFA<0THENA=A+360
680 INPUT"ENTER RANGE 2 - 20";R:IFR<2THEN760ELSEIFR>20THENR=RND(
10)+5
690 IFA>315ORA<45OR(A>135AND(A<225)THENR=INT(R/3)*2:IFR<2THENR=2
700 S1=0:H=2:GOSUB50
710 PRINT@704,A$"ENTER BEARING FOR 4-INCH GUNFIRE FROM "A$(X);:I
NPUTA
720 IFA<0THENA=A+360:GOTO720ELSEIFA>360A=A-360:GOTO720
730 A=A-A(X,1):INPUT"ENTER RANGE 2 - 10";R:IFR<2THEN760ELSEIFR>1
0THENR=RND(5)+5
740 IFA>315ORA<45OR(A>135AND(A<225)THENR=INT(R/3)*2:IFR<2THENR=2
750 H=1:S1=1:GOSUB50
760 NEXTX

```

Topedo fire.

```

770 FORX=8TO16:PRINT@640,A$;:IFA(X,3)<=0DRA(X,9)<3THEN820
780 PRINT"ENTER BEARING FOR TORPEDOE FIRE FROM "A$(X);:INPUTA:IF
A=999THEN820

```

```

790 IFA<0THENA=A+360:GOTO790ELSEIFA>360THENA=A-360:GOTO790
800 H=3:S1=RND(5)+9
810 FORR=2TOS1:GDSUB50:NEXTR
820 NEXTX

```

Japanese movement.

```

830 PRINT@640,A#"NOW MOVING JAPANESE SHIPS"
840 IFJ(0,4)>30ANDRND(2)=1THENREADJ,M
850 READJ,M:IFJ=999PRINT@640,"GAME OVER- I WILL NOW TABULATE THE
SCORE":GOTO1480
860 FORX=0T07:RESET(J(X,4),J(X,5)):NEXTX
870 FORX=7T01STEP-1:J(X,4)=J(X-1,4):J(X,5)=J(X-1,5)+RND(2)-2:IFJ
(X,6)=-1ANDJ(X,3)>0THENSET(J(X,4),J(X,5)):NEXTXELSENEXTX
880 M=M+RND(3)-2:J(0,4)=J:J(0,5)=M:IFJ(0,6)THENSET(J,M)

```

Allied movement.

```

890 BB=0:PRINT@640,A#"NOW MOVING ALLIED SHIPS":FORX=0T016:IF6(X)
=0THENIFA(X,6)>=0ANDA(X,6)<=127ANDA(X,7)>=0ANDA(X,7)<29THENRESET
(A(X,6),A(X,7)):IFA(X,3)<=0THEN1070
900 IFA(X,3)<=0THEN1070ELSEONA(X,4)GOTO910,920,930,940,950,960,9
70,980
910 H=0:V=-1*A(X,5):GOTO1000
920 H=A(X,5):V=-1*A(X,5):GOTO990
930 H=A(X,5):V=0:GOTO1000
940 H=A(X,5):V=A(X,5):GOTO990
950 H=0:V=A(X,5):GOTO1000
960 H=-1*A(X,5):V=A(X,5):GOTO990
970 H=-1*A(X,5):V=0:GOTO1000

```



SAVO ISLAND

```

980 H=-1*A(X,5):V=-1*A(X,5)
990 H=INT((H+1)/2):V=INT((V+1)/2)
1000 A(X,6)=A(X,6)+H:A(X,7)=A(X,7)+V:H=A(X,6):V=A(X,7)
1010 IFH<0ORV<0ORH>127ORV>29THENA(X,3)=0:PRINT@640,A$:A$(X)" HAS
LEFT THE BATTLE AREA HER CAPTAIN WILL BE COURT MARSHALL
ED LATER, FOR NOW HOWEVER WE CAN CONSIDER HER SUNK":FORY=1TO1000
:NEXTY:GOTO1070
1020 IFPOINT(H,V)AND(POINT(H+1,V)ORPOINT(H-1,V))PRINT@640,A$:A$(
X)" JUST RAN AGROUND":A(X,3)=0:G(X)=-1
1030 FORZ=0TO7:IFH=J(Z,4)ANDV=J(Z,5)PRINT@640,A$:A$(X)" JUST COL
LIDED WITH "J$(Z):A(X,3)=A(X,3)-5:J(Z,3)=J(Z,3)-2:NEXTZELSENEXTZ
1040 FORZ=1TO100:NEXTZ:FORZ=0TO16:IFZ=XORA(Z,3)<=0ORA(X,3)<=0THE
N1060
1050 IFA(Z,6)=HANDA(Z,7)=VPRINT@640,A$:A$(X)" JUST COLLIDED WITH
"A$(Z):A(X,3)=A(X,3)-5:A(Z,3)=A(Z,3)-5:FORS2=1TO100:NEXTS2
1060 NEXTZ:IFA(X,3)>0THENSET(H,V)
1070 NEXTX

```

Japanese fire.

```

1080 IFJ(0,4)<40THENFORX=0TO7:IFNOTJ(X,6)THENNEXTX:GOTO1200
1090 PRINT@640,A$"INCOMING JAPANESE FIRE"
1100 FORX=0TO7:B(X)=0:PRINT@704,A$:IFJ(X,3)<=0THEN1190
1110 FORY=0TO16:IFA(Y,3)<=0THEN1180
1120 IFA(Y,6)>J(X,4)+15ORA(Y,6)<J(X,4)-15THEN1180
1130 IFA(Y,7)>J(X,5)+14ORA(Y,7)<J(X,5)-14THEN1180
1140 D=RND(10)+J(X,0)-A(Y,5)-A(Y,9):IFD>4PRINT@704,"STRATTLED "A
$(Y):FORZ=1TO500:NEXTZ:A(Y,3)=A(Y,3)-(2+RND(3))
1150 IFD>7PRINT@704,"DIRECT HIT ON "A$(Y):FORZ=1TO500:NEXTZ:A(Y,
3)=A(Y,3)-(2+RND(4)):IFRND(3)=1THENA(Y,0)=A(Y,0)-RND(2)
1160 D=RND(10)+J(X,0)-A(Y,5)-A(Y,9):IFD>6PRINT@768,"TORPEDOE HIT
ON "A$(Y):FORZ=1TO500:NEXTZ:A(Y,3)=A(Y,3)-(3+RND(5)):Z=-1:IFRND
(2)=1THENA(Y,0)=A(Y,0)-RND(2)
1170 B(X)=-1:IFZTHENY=17:Z=0
1180 NEXTY
1190 NEXTX

```

Update status of all ships.

```

1200 PRINT@640,A$"NOW UPDATING THE STATUS OF ALL SHIPS"
1210 FORX=0TO16
1220 IFA(X,3)<=0THENA(X,3)=0:IFG(X)=0THENIFA(X,6)>0ANDA(X,6)<127
ANDA(X,7)>0ANDA(X,7)<47THENRESET(A(X,6),A(X,7))
1230 IFX>7THEN1250
1240 IFJ(X,3)<=0THENJ(X,3)=0:J(X,6)=0
1250 NEXTX:FL=0
1260 FORX=0TO16:IFA(X,3)<=0THEN1320
1270 FL=-1:FORY=0TO7:IFJ(Y,3)<=0THEN1310
1280 IFJ(Y,4)<A(X,6)-5ORJ(Y,4)>A(X,6)+5THEN1310
1290 IFJ(Y,5)<A(X,7)-4ORJ(Y,5)>A(X,7)+4THEN1310

```

```

1300 PRINT#640,A$;A$(X)" REPORTS SIGHTING UNIDENTIFIED SHIPS";A$
:B(Y)--1:FORZ=1TO150:NEXTZ
1310 NEXTY
1320 NEXTX

```

Check for any Japanese ships afloat.

```

1330 JF=0:FORX=0TO7:IFJ(X,3)>0THENJ(X,6)=B(X):JF=-1:NEXTXELSENEXTX
1340 IFFLANDJFTHEN560

```

No Japanese ships are afloat, so go to the winning message.

```

1350 IFNOTJFTHEN1590

```

Sorry, you lost.

```

1360 CLS:PRINT"IF I GO AWAY TO SEA,":PRINTTAB(16);"I SHALL RETURN A CORPSE AWASH,"
1370 PRINT"IF DUTY CALLS ME TO THE MOUNTAINS,":PRINTTAB(16);"A BURDENED SWATH WILL BE MY PAWL,"
1380 PRINT"THUS FOR THE SAKE OF THE EMPEROR,":PRINTTAB(16);"I WILL NOT DIE PEACEFULLY AT HOME!":PRINT
1390 PRINT"THESE WORDS FROM THE ANCIENT SONG OF THE SAMURI":PRINT"ARE TO TELL YOU THAT YOU HAVE LOST THE BATTLE OF SAVO ISLAND"
1400 PRINT:PRINT"HIT ANY KEY TO CONTINUE"
1410 C$=INKEY$:IFC$=""THEN1410
1420 CLS:PRINT"THE RESULTS OF THE ACTUAL BATTLE WERE:":PRINTTAB(13);"ALLIED LOSSES:":PRINT"3 AMERICAN CRUISERS SUNK":PRINT"1 AUSTRALIAN CRUISER SUNK":PRINT"1 AMERICAN CRUISER BADLY DAMAGED":PRINT"2 DESTROYERS SUNK":PRINT"3 DESTROYERS DAMAGED"
1430 PRINT"1,270 OFFICERS AND MEN KILLED":PRINT"709 WOUNDED":PRINT"JAPANESE LOSSES:":PRINT"37 MEN KILLED":PRINT"57 WOUNDED":PRINT"NO DAMAGE TO ANY SHIP."
1440 PRINT"HIT ANY KEY TO CONTINUE"
1450 C$=INKEY$:IFC$=""THEN1450
1460 CLS:PRINT"THE NEXT DAY THE U.S. SUB S-44 PUT A TORPEDOE INTO THE KAKO, SMALL PRICE FOR THE MOST HUMILIATING DEFEAT EVER SUFFERED BY THE U.S. NAVY IN A FAIR FIGHT."

```

Compute the final score. This routine does not execute if all the Japanese ships sank.

```

1470 AT=0:JT=0:PRINT:PRINT"1 AM NOW TOTALING THE SCORE"
1480 FORX=0TO16:IFX<8THENY=A(X,3)/30ELSEY=A(X,3)/15
1490 IFA(X,3)=0THENJT=JT+100
1500 AS=Y*A(X,8):JS=A(X,8)-AS:AT=AT+AS:JT=JT+JS:NEXTX
1510 FORX=0TO6:Y=J(X,3)/30:JS=Y*J(X,7):AS=J(X,7)-JS:JT=JT+JS:AT=AT+AS:IFJ(X,3)=0THENAT=AT+200
1520 NEXTX:Y=J(7,3)/15:JS=Y*J(7,7):AS=J(7,7)-JS:JT=JT+JS:AT=AT+AS
1530 PRINT"YOU SCORED"AT"POINTS":PRINT"THE JAPANESE (COMPUTER) SCORED"JT"POINTS"
1540 IFAT>JTTHENR$="TACTICAL ALLIED VICTORY"

```


SAVO ISLAND



```

1550 IFJT>=ATTHENR$="TACTICAL JAPANESE VICTORY"
1560 IFJT>2*ATTHENR$="STRATEGIC JAPANESE VICTORY"
1570 IFAT>2*JTTHENR$="STRATEGIC ALLIED VICTORY"
1580 PRINT"THAT COMPUTES TO A "R$:PRINT:PRINT"TO PLAY AGAIN TYPE
RUN":END

```

You win!

```

1590 CLS:PRINT"I WILL CALMLY OFFER UP THE COURAGE OF MY SOUL,":P
RINTAB(11);"I WILL NEVER SUFFER THE DISGRACE OF BEING TAKEN ALI
VE":PRINT
1600 PRINT"-FROM THE CODE OF THE JAPANESE WARRIOR-":PRINT
1610 PRINT"SOMEHOW YOU MANAGED TO COMPLETELY DESTROY THE JAPANESE
FLEET":PRINT"WHEN DID YOU ATTEND ANNAPOLIS ? "
1620 PRINT:PRINT"TO PLAY AGAIN TYPE RUN":END

```

Data.

```

1630 DATA "NORTH","NORTH-EAST","EAST","SOUTH-EAST"
1640 DATA "SOUTH","SOUTH-WEST","WEST","NORTH-WEST"
1650 DATA "AUSTRALIA",8,4,0,30,1,1,115,21,200,2
1660 DATA "CHICAGO", 9,8,0,30,3,1, 64,23,300,2
1670 DATA "CANBERRA", 8,4,0,30,3,1, 72,23,200,2
1680 DATA "VINCENNES",9,8,0,30,3,1, 70, 5,300,2
1690 DATA "QUINCY", 9,8,0,30,3,1, 66, 5,300,2
1700 DATA "ASTORIA", 9,8,0,30,3,1, 61, 5,300,2
1710 DATA "SAN JUAN", 8,4,0,30,1,1,102,14,200,2
1720 DATA "HOBART", 8,4,0,30,1,1,102,17,200,2
1730 DATA "PATTERSON",0,4,4,15,3,1, 68,21,100,2
1740 DATA "BAGLEY", 0,4,4,15,3,1, 68,24,100,2
1750 DATA "BLUE", 0,4,4,15,5,3, 25,17,100,4
1760 DATA "HELM", 0,4,4,15,3,1, 74, 4,100,2
1770 DATA "WILSON", 0,4,4,15,5,1, 75, 7,100,2
1780 DATA "RALPH TALBOT",0,4,4,15,1,3,25,12,100,4
1790 DATA "BUCHANAN",0,4,4,15,1,1,100,12,100,2
1800 DATA "MONSSEN",0,4,4,15,1,1,105,12,100,2
1810 DATA "JARVIS", 0,2,2,9,7,3,111,19,50,1
1820 DATA "CHOKAI",10,4,8,30,7,12,0,400

```

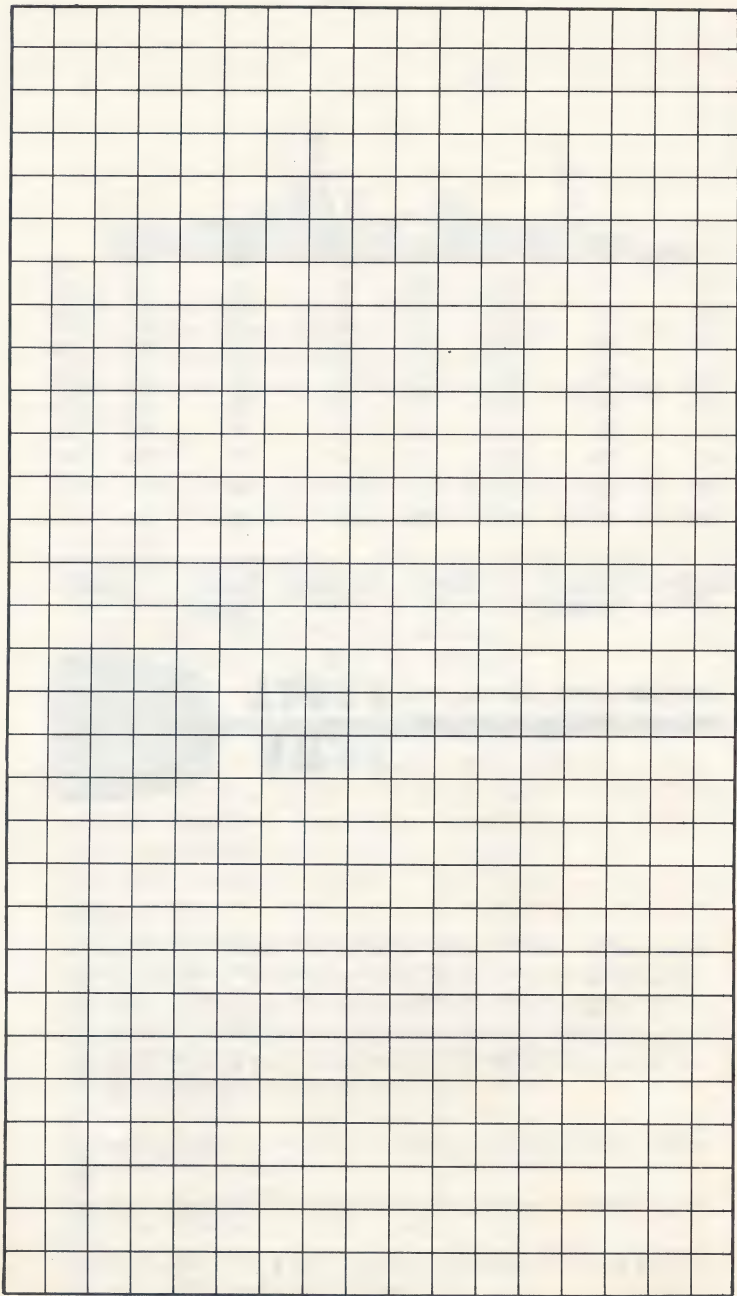
1830 DATA "AOBA",6,4,4,30,6,11,0,400
 1840 DATA "KAKO",6,4,4,30,5,10,0,400,"KINUGASA",6,4,4,30,4,9,0,4
 00
 1850 DATA "FURUTAKA",6,4,4,30,3,8,0,400,"TENRYU",0,4,6,30,2,7,0,
 350
 1860 DATA "YUBARI",0,6,4,30,1,7,0,300,"YUNIGI",3,0,4,15,0,7,0,20
 0
 1870 DATA 12,13,18,15,21,15,27,14,30,15
 1880 DATA 36,15,39,17,42,19,45,19,48,19,51,19,54,19,57,19,60,19
 1890 DATA 63,19,66,19,69,19,72,19,75,19,78,17,81,16,82,13,79,11
 1900 DATA 76,11,73,10,70,10,67,10,64,10,61,10,58,10,55,07,52,07
 1910 DATA 49,06,46,06,43,06,40,06,37,06,34,06,31,06,28,06,25,05
 1920 DATA 22,05,19,05,16,05,13,05,10,05,7,05,4,05,1,05,999,999,9
 99,999



SWAT TABLE For TRS-80® SAVO ISLAND

LINES	SWAT CODE	LENGTH	LINES	SWAT CODE	LENGTH
10 - 140	NR	429	1020 - 1100	MB	508
150 - 220	MV	505	1110 - 1200	XF	534
230 - 310	XK	537	1210 - 1320	HU	399
320 - 400	HZ	533	1330 - 1420	XW	701
410 - 490	RU	525	1430 - 1500	BB	522
500 - 530	QR	527	1510 - 1590	ZF	570
540 - 650	HU	563	1600 - 1690	KH	501
660 - 770	BT	479	1700 - 1810	NZ	526
780 - 890	LA	600	1820 - 1910	RC	534
900 - 1010	TX	545	1920 - 1920	WR	67

Map Your Adventure Here





SoftSide



CV/DV Adventure Series

"In this adventure, I will become your eyes, ears, and hands," says your computer. You are about to enter a new fantasy world. Each issue, *SoftSide* DV and CV present the latest challenge to your ingenuity and perseverance. For those unfamiliar with the genre, the fantasy/adventure game places you in a puzzling situation, usually in a strange, unfamiliar world, but sometimes in a world enough like your own to lull you into a false sense of security. Your first goal is, often, simply to survive. However, success at even this basic task can be doubtful. Perplexing situations will certainly test your ingenuity and perseverance, and perhaps you will glean great treasures. But dragons and desperadoes may oppose you — you never know.

To "win" a fantasy/adventure game, you have to solve the puzzles and overcome the obstacles that confront you. Death is transitory — you can always re-run the program. Aficionados of adventures carefully map the locations in the game's world. If you have an exceptional memory, you may skip this exercise... Now, was the cave with the ruby-encrusted scepter north or east of the beach? Hmmm...

You act by giving your computer simple, one- or two-word commands, like "LOOK", or "GET RUBY". The introduction to each adventure explains this more fully.

One issue after the appearance of an adventure, *SoftSide* will publish encrypted hints for it. The encryption will prevent you from inadvertently seeing the hints, but will be simple enough to permit easy reading if an adventure truly stumps you.

To begin the adventure, just RUN the program named "INTRO" on your disk, or select the adventure from the DV menu. On cassette, the adventure is the last program, and the INTRO program immediately precedes it.

Memory requirements for all adventures — 16K tape, 32K disk.



General Information

These are the standard procedures for the programs published in **SoftSide Selections**. Sometimes, a particular program does not lend itself to these procedures. Always read the specific instructions accompanying a program. They will instruct you if there are any variances from the following procedures. Also, back issues of **SoftSide Magazine** may differ in some details.



SWAT TABLE

At the conclusion of each program listing in **SoftSide Selections**, we include a **SWAT (Strategic Weapon Against Typos) Table**. **SWAT** for the TRS-80 appeared in **SoftSide** Issue #30. If you missed Issue #30, we'll send you a free reprint of **SWAT**. Send a self-addressed, stamped envelope to:

SoftSide Publications, Inc.
Department **SWAT**
6 South Street
Milford, NH 03055

Please be sure to tell us that you have a TRS-80 computer.

Magnetic Media

Disks are available in Model I or Model III format. They contain the DOSPLUS operating system. A cover program runs automatically when you boot the disk. Back issues earlier than May 1981 are available only in Model I format. If you have a two-drive Model III, you can convert such disks with the CONVERT utility.

Tapes CLOAD in the normal manner on Model I's, and at low speed (500 baud) on Model III's. The first program is a cover/menu program; side two of the tape is a duplicate of side one.

SoftSide Selections disks and tapes are duplicated on reliable, professional equipment. Bad copies are exceedingly rare. Nevertheless, the trip through the mail occasionally results in damage to the sensitive magnetic media. If, after a reasonable number of attempts on well-adjusted, clean equipment, you are unable to load a program, return it to us along with an exact explanation of your problem. We will send you a replacement copy.

SoftSide Selections media are not copy protected. We urge you to make an archival backup copy of your disk or tape as soon as you receive it, as our replacement policy is valid only for 30 days. Please resist the urge to give away copies of copyrighted material.

Line Listings

The line listings in this booklet are in standard 64-column format, and they appear exactly as they should on your screen when you type LIST.

System Requirements

The necessary memory and other equipment you need to run a program are listed in the introductory paragraph of the article for each program. (Also see the **SoftSide Adventure Series** elsewhere in this booklet.)



Notes



Tired of Typing?

- Use And Enjoy These Programs Right Away!
- Avoid Worry About Typos!
- Get Additional Programs to Enjoy Without Typing.



Order this Issue's Programs on Disk or Cassette!

Send the coupon below to:

SoftSide Publications, Inc.
6 South Street
Milford, NH 03055



CV DV

SoftSide™ Selections

ORDER THIS ISSUE'S MEDIA

Having difficulty finding the time to input the programs from this issue of **SoftSide Selections**? Don't despair — the cassette and disk versions are still available! **And**, each DV (disk version) contains an additional program.

☐ #39 DV \$16.00 ☐ #39 CV \$8.00

I own an ☐ Apple® ☐ IBM® PC (Disk Only - No Enhancement)
☐ Atari® ☐ TRS-80® Mod. I ☐ TRS-80® Mod. III

Name

Address

City/State Zip

☐ **Check or Money Order**

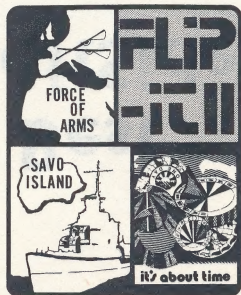
☐ MasterCard ☐ VISA

Name of Cardholder

MC# and Interbank#/VISA#

Exp. Date Signature

SoftSide Selections



A product of SoftSide Publications, Inc.
6 South Street, Milford, NH 03055.

Prices subject to change without notice Apple®, IBM®, Atari® and TRS-80® are registered trademarks of The Apple Computer Company, International Business Machines, Inc., Warner Communications, and The Tandy Corporation respectively.

Like SoftSide? Wait 'til you've seen our **BEST!**



The Best of SoftSide!

For the past four years, **SoftSide Magazine** has been bringing Apple®, Atari®, and TRS-80® owners the best in BASIC software. But now you can do even better...**The Best of SoftSide**. From all our back issues, we've selected the most useful...the most entertaining...the most fun programs **SoftSide** has ever published.

The Best of SoftSide is available in three versions...one for Apple, Atari, and TRS-80. Each contains page after page of BASIC code for adventures, simulations, practical applications, and much more.

To order your copy of **The Best of SoftSide**, fill out the coupon below, and mail it along with \$19.95 to **SoftSide**, 6 South Street, Milford, New Hampshire 03055.

The Best of SoftSide

☐ **YES...** I want to enjoy the best programs — Games, Utilities, Adventures — from **SoftSide Magazine's** past. Please rush my copies of **The Best of SoftSide** to:

Name

Address

City State Zip

Here's my order For The Best of SoftSide

Computer	Quantity Book Only \$19.95 ea.	Quantity Book plus Disk \$68.95 ea.
Apple®		
Atari®		
TRS-80®		
TOTAL COPIES	@\$19.95	@\$68.95



☐ I've already ordered **The Best of SoftSide** book. I'm enclosing \$49 for the disks only.

Total Enclosed \$

I'm paying by: ☐ Check ☐ Money Order
☐ VISA ☐ MasterCard

(Foreign order please include \$5 postage. PAY IN USA FUNDS ONLY.)

VISA/MasterCard# and Interbank# Exp. Date

Name of Cardholder

Signature



SoftSideTM Selections

- Challenge your computer's formidable electronic intellect in **Flip-It II**, SoftSide's version of Reversi.

- As the commander of the Allied fleet, you can change history in the battle of **Savo Island**.

- **DV Bonus: Force of Arms** — you must conquer all of the Earth's territories before your opponents do, while the computer keeps track of all the rules. Who's the craftiest general on **your** block?

- **Adventure: It's About Time** — The future of the Earth is at stake, and you must use your Time Machine to prevent the evil Henry Bowman from using his apocalyptic B Bomb.

SOFTSIDE PUBLICATIONS, INC.

Milford, New Hampshire